Cue and Sensor Fusion for Independent Moving Objects Detection and Description in Driving Scenes

Nikolay Chumerin and Marc M. Van Hulle

Katholieke Universiteit Leuven, Laboratorium voor Neuro- en Psychofysiologie, Campus Gasthuisberg, Herestraat 49, bus 1021, B-3000 Leuven, Belgium
{nikolay.chumerin, marc.vanhulle}@med.kuleuven.be

Summary. In this study we present an approach to detecting, describing and tracking independently moving objects (IMOs) in stereo video sequences acquired by on-board cameras on a moving vehicle. In the proposed model only three sensors are used: stereovision, speedometer and LIDAR (Light Detection and Ranging). The IMOs detected by vision are matched with obstacles provided by LIDAR. In the case of a successful matching, the descriptions of the IMOs (distance, relative speed and acceleration) are provided by ACC (Adaptive Cruise Control) LIDAR sensor, or otherwise these descriptions are estimated based on vision. Absolute speed of the IMO is evaluated using its relative velocity and egospeed provided by the speedometer. Preliminary results indicate the generalization ability of the proposed system.

1 Introduction

The detection of the *independently moving objects* (IMOs) can be considered as an exponent of the obstacle detection problem, which plays a crucial role in traffic-related computer vision. Vision alone is able to provide robust and reliable information for autonomous driving or guidance systems in real time but not for the full spectrum of real world scenarios. The problem is complicated by ego-motion, camera vibrations, imperfect calibrations, complex outdoor environments, insufficient camera resolutions and other limitations. The fusion of information obtained from multiple sensors can dramatically improve the detection performance [11, 29, 2, 4, 3, 8, 16, 28, 18, 12, 15, 5, 27, 17, 9, 30].

In Table 1 we present a chronological list of studies which are related to sensor fusion in traffic applications and which are relevant to the considered topic. Various sensors can be used for traffic applications: video (color or grayscale) cameras in different setups (monocular, binocular or trinocular), IR (infrared) cameras, LIDAR (Light Detection and Ranging), radar (Radio Detection and Ranging), GPS/DGPS (Global Positioning System/Differential GPS) as well as data from vehicle IMU (Inertial Measurement Unit) sensors:

accelerometer, speedometer, odometer and angular rate sensors (gyroscopes). There are a number of approaches to fusion characterization [10, 7, 24, 33] but, most frequently, fusion is characterized by the abstraction level:

1. Low (signal) level fusion combines raw data provided directly from sensors, without any preprocessing or transformation.
2. Intermediate (feature) level fusion aggregates features (e.g. edges, corners, texture) extracted from raw data before aggregation.
3. High (decision) level fusion aligns decisions proposed by different sources.

Depending on the application, several different techniques are used for fusion. Matching of the targets detected by different sensors is often used for obstacle detection. Extensions of the Kalman filter (KF) [14](e.g. extended Kalman filter (EKF) and unscented Kalman filter (UKF) [13]) are mostly involved in estimation and tracking of obstacle parameters, as well as in ego-position and ego-motion estimation.

The flow diagram of the proposed model is shown on Fig. 1. In order to detect independent motion we propose to extract visual cues and subsequently fuse them using appropriately trained multi-layer perceptron (MLP). Object recognition is used as cooperative stream which helps to delineate and classify IMOs. If detected by vision IMO appears within sweep of the ACC LIDAR, we use distance, speed and acceleration of the IMO provided by ACC system. Otherwise these descriptions are estimated based on vision.

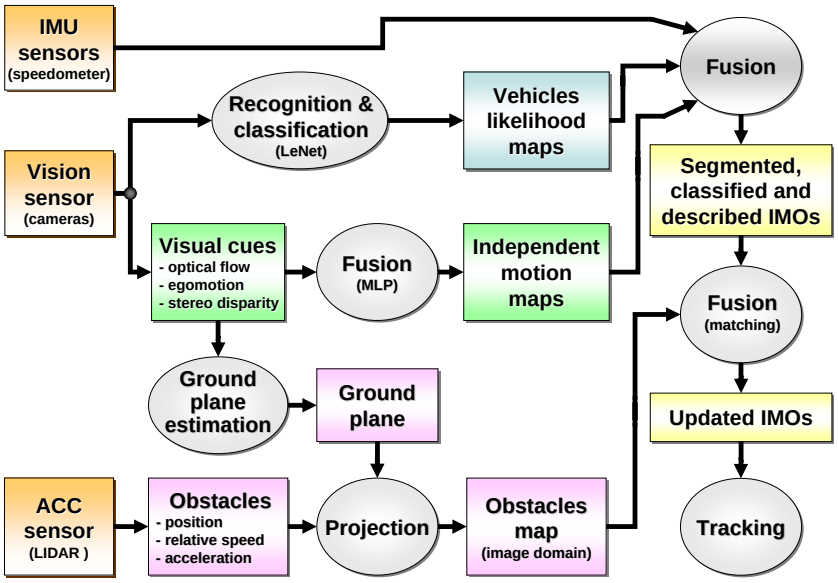

Fig. 1. Flow-diagram of the proposed model.

Table 1. Sensor fusion for traffic applications papers short overview

Study	Sensors	Cues/Features	Techniques used
Handmann *et al.* [11]	monocular color vision, radar	color, edges, texture (local image entropy), (up to 3) obstacle positions	MLP
Stiller *et al.* [29]	stereo vision, radar, LIDARs, DGPS/INS	horizontal edges, stereo disparity, optical flow, 2D range profile, global ego-position and ego-orientation	Kalman filter
Becker and Simon [2]	stereo vision, DGPS, vehicle sensors (IMUs), LIDARs, radar	local ego-position and ego-orientation (w.r.t. lane), global ego-position and ego-orientation, egospeed, egoacceleration, steering angle, 2D range profile	Kalman filter
Kato *et al.* [16]	monocular vision, radar	Kanade-Lucas-Tomasi feature points, range data	frame-to-frame feature points coupling based on range data
Fang *et al.* [8]	stereo vision, radar	edges, stereo disparity, depth ranges	depth-based target edges selection and contour discrimination
Steux *et al.* [28]	monocular color vision, radar	shadow position, rear lights position, symmetry, color, 2D range profile	belief network
Hofmann *et al.* [12]	monocular color vision, monocular grayscale vision, radar, ACC-radar	lane position and width, relative ego-position and ego-orientation (w.r.t. road), radar-based obstacles	extended Kalman filter
Laneurit *et al.* [18]	vision, GPS, odometer, wheel angle sensor, LIDAR	relative ego-position and ego-orientation (w.r.t. road), global ego-position and ego-orientation, steering angle, path length, LIDAR-based obstacle profile	Kalman filter
Sergi [26]	vision, LIDAR, DGPS	video stream, global ego-position and ego-orientation, LIDAR-based obstacle profile	Kalman filter
Sole *et al.* [27]	monocular vision, radar	horizontal and vertical edges, 'pole like' structures, radar target,	matching
Blanc *et al.* [5]	IR camera, radar, LIDAR	IR images, range profile	Kalman filter, matching
Labayrade *et al.* [17]	stereo vision, LIDAR	stereo disparity, "v-disparity", lighting conditions, road geometry, obstacle positions	matching, Kalman filter, belief theory based association
Thrun *et al.* [31]	monocular color vision, GPS, LIDARs, radars, accelerometers, gyroscopes	color images, global ego-position and ego-orientation, egospeed, short-range profile (LIDARs), long-range obstacles (radars)	Unscented Kalman Filter

In order to validate the model we have used the data obtained in the frameworks of the DRIVSCO and ECOVISION European Projects. In recording sessions a modified Volkswagen Passat B5 was used as a test car. It was equipped by Hella KGaA Hueck & Co.

2 Vision sensor data processing

For vision-based IMO detection, we used an approach proposed by Chumerin and Van Hulle [6]. This method is based on the processing and subsequent fusing of two cooperative streams: the *independent motion detection stream* and the *object recognition stream*. The recognition stream deals with static images (*i.e.*, does not use temporal information) and therefore cannot distinguish between independently moving and static (*i.e.*, with respect to the environment) objects, but which can be detected by the independent motion stream.

2.1 Vision sensor setup

In the recording sessions, we used a setup with two high resolution progressive scan color CCD cameras (see Table 2). The camera rig was mounted inside the cabin of the test car (see Fig. 2) at 1.24 m height above the ground, with 1.83 m from the front end and 17 cm displacement from the middle of the test car towards the driver's side. Both cameras were oriented parallel to each other and to the longitudinal axis of the car and look straight ahead into the street. Before each recording session, the cameras were calibrated. Raw color (Bayer pattern) images and some crucial ACC parameters were stored via CAN-bus for further off-line processing. In the model, we used rectified grayscale images downscaled to a 320×256 pixels resolution.

Table 2. Video sensor specifications

Sensor parameter	Value
Manufacturer	JAI PULNiX Inc.
Model	TMC-1402Cl
Field of View	$53^\circ \times 42.4^\circ$ (horizontal×vertical)
Used resolution	1280×1024
Used frequency	25 fps
Color	RGB Bayer pattern
Interocular distance	330 mm
Focal length	12.5 mm
Optics	Pentax TV lenses

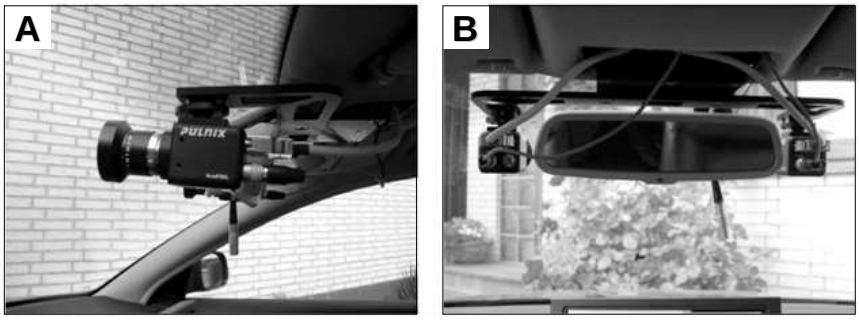

Fig. 2. Setup of the cameras in the car.

2.2 Independent motion stream

The problem of *independent motion* detection can be defined as the problem of locating objects that move independently from the observer in his field of view. In our case, we build so-called *independent motion maps* where each pixel encodes the likelihood of belonging to an IMO. For each frame we build an independent motion map in two steps: visual cues extraction and classification.

As visual cues we consider: *stereo disparity* (three components – for current, previous and next frame), *optical flow* (two components) and *normalized coordinates*[1] (two components). The optical flow and stereo disparity are computed using multiscale phase-based optical flow and stereo disparity algorithms [23, 25]. Unfortunately, there are no possibilities to estimate reliably all these cues for every pixel in the entire frame. This means that the motion stream contains incomplete information, but this gap will be bridged after fusion with the recognition stream.

We consider each pixel as a multidimensional vector with visual cues as components. We classify all the pixels (which have every component properly defined) in two classes: IMO or background. We have tried a number of setups for classification, but the optimal performance was obtained with a multi-layer perceptron (MLP) with three layers: a linear (4–8 neurons), a nonlinear layer (8–16 neurons), and one linear neuron as output.

For training purposes, we labeled the pixels in every frame of a number of movies into background and different IMOs, using a propriety computer-assisted labeling tool (see Fig. 4).

After training, the MLP can be used for building an IMO likelihood map I for the entire frame:

$$I(x, y) = p\left(IMO|(x, y)\right), \tag{1}$$

[1] By a normalized coordinate system on a frame we mean the rectangular coordinate system with origin in the center of the frame, where the upper-left corner is $(-1, -1)$ and the lower-right corner is $(1, 1)$.

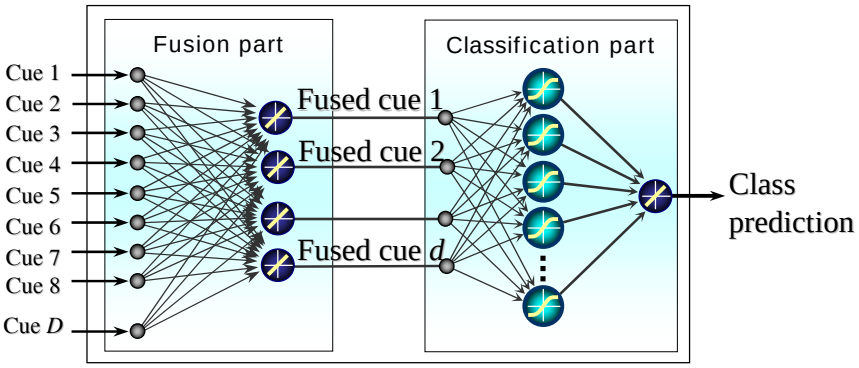

Fig. 3. MLP used as classifier in independent motion stream.

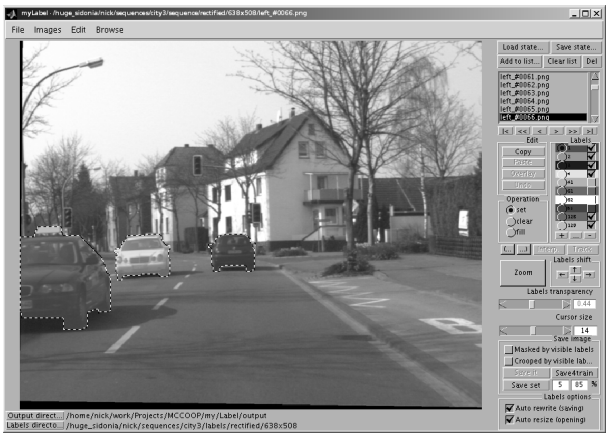

Fig. 4. myLabel – a tool for labeling video sequences. The labeling is similar to what is done in graphical editors like Photoshop or Gimp. Each label mask is represented by a separate layer with its own color (on the figure instead of colors we have used contours). The user can easily edit label masks in a pixel-wise manner as well as change their colors, transparencies and visibilities. myLabel allows semi-automatic labeling by interpolating the labels between two labeled frames.

where x, y are pixel coordinates. Fig. 5 shows an example of a IMO likelihood map obtained using the proposed approach.

2.3 Recognition stream

For the recognition of vehicles and other potentially dangerous objects (such as bicycles and motorcycles, but also pedestrians), we have used a state of the art recognition paradigm – the convolutional network LeNet, proposed by LeCun and colleagues [19]. We have used the CSCSCF configuration of LeNet (see Fig. 6) comprising six layers: three convolutional layers (C0, C1, C2), two

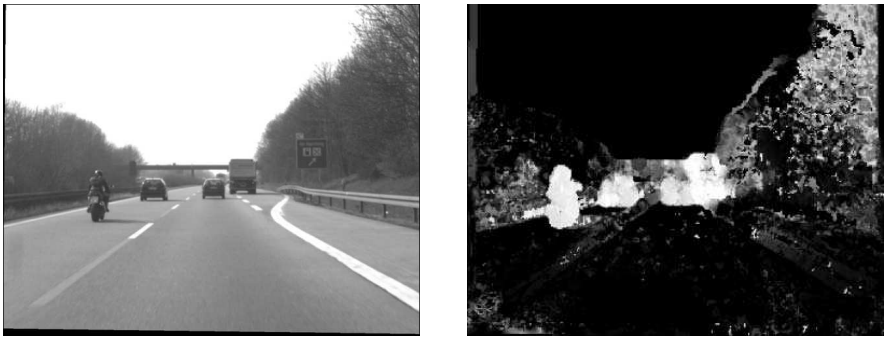

Fig. 5. (Left) Frame number 342 of motorway3 sequence. (Right) Matrix I, output of the motion stream for the same frame. Value $I(x, y)$ is defined as probability of pixel (x, y) being part of an IMO.

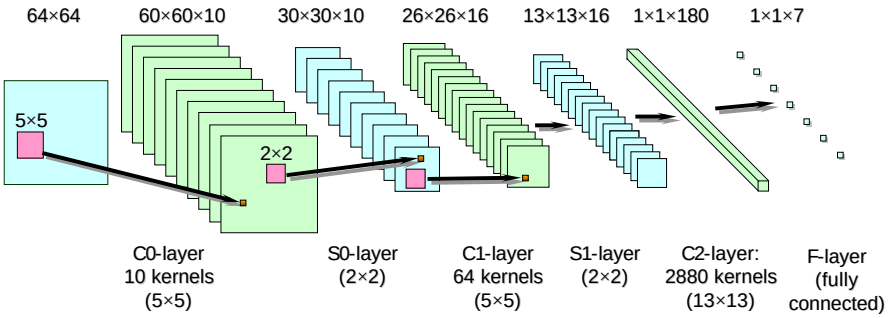

Fig. 6. LeNet – a feed-forward convolutional neural network, used in the recognition stream.

subsampling layers (S0, S1) and one fully connected layer (F). As an input, LeNet receives a 64×64 grayscale image. Layer C0 convolves the input with ten 5×5 kernels, adds (ten) corresponding biases, and passes the result to a squashing function[2] to obtain ten 60×60 feature maps.

In layer S0, each 60×60 map is subsampled to a 30×30 map, in such a way that each element of S0 is obtained from a 2×2 region of C1 by summing these four elements, by multiplying with a coefficient, adding a bias, and by squashing the end-result. For different S0 elements, the corresponding C1's 2×2 regions do not overlap. The S0 layer has ten coefficient-bias couples (one couple for each feature map). Computations in C1 are the same as in C0 with the only difference in the connectivity: each C1 feature map is not obtained by a single convolution, but as a sum of convolutions with a set of previous (S0) maps (see Table 3). Layer S1 subsamples the feature maps of C1 in the same manner as S0 subsamples the feature maps of C0. The final

[2] $f(x) = A \tanh(Sx)$, $A = 1.7159$ and $S = 2/3$ according to [19].

convolutional layer C2 has kernels sized 13×13 and 180 feature maps which are fully connected to all 16 S1 feature maps. It means that the number of C2 kernels is $16 \times 180 = 2880$, and the corresponding connectivity matrix should have all cells shaded. The output layer consists of seven neurons, which are fully connected to C2's outputs. It means that each neuron in F (corresponding to a particular class *background*, *cars*, *motorbikes*, *trucks*, *buses*, *bicycles* and *pedestrians*) just squashes the biased weighted sum of all C2's outputs.

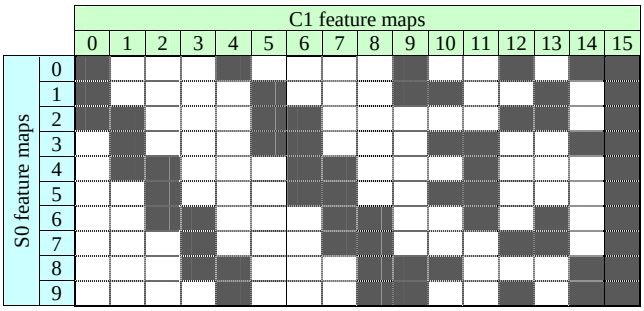

Table 3. S0-C1 connectivity matrix. A shaded cell which belongs to the i-th column and j-th row indicates that the j-th feature map of S0 participates in the computation of the i-th feature map of C1. For example, to compute the fourth feature map of C1, one has to find a sum of convolutions of S0 feature maps 0, 8 and 9 with corresponding kernels. The number of kernels in C1 (the number of shaded cells in the table) is 64.

LeNet scans (see Fig. 7) the input image with a sliding window and builds seven matrices, $R_0, \ldots, R_6$, which are regarded as likelihood maps for the considered classes. In order to make the recognition more scale-invariant, we process the input image in two scales 320×256 and 640×512, but the resulting (mixed) matrices R_i are sized 320×256. Note that, for further processing, the most important map is R_0, which corresponds to the background class and the so-called *non-background* map is obtained as $(1 - R_0)$. The rest of the maps $(R_1, \ldots, R_6)$ are used only in IMO classification.

2.4 Training

For training both vision streams, we used two rectified stereo video sequences, each consisting of 450 frames. We have labeled IMOs in all left frames of the sequences. These labels were used for training the motion stream classifier.

We have used small batches with the increasing size version of the BFGS Quasi-Newton algorithm for the independent motion classifier training. Samples for each batch were randomly taken from all the frames of all the scenes. Training was stopped after reaching 0.04 (MSE on training set) performance.

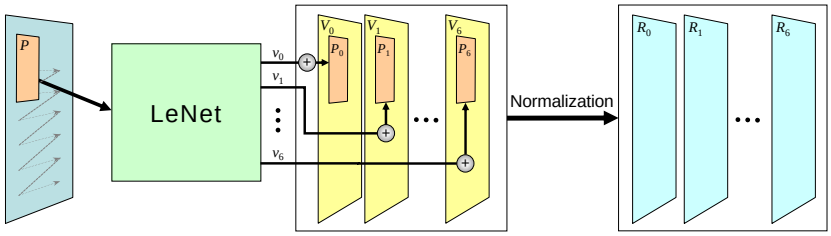

Fig. 7. For each position of the 64×64 sliding window we feed the corresponding image patch P to LeNet in order to obtain seven values $v_0, \ldots, v_6$ which are related to the class likelihoods of P. Then we update the regions P_i associated with P in intermediate matrices V_i by adding v_i $(i = 0, \ldots, 6)$. After scanning the entire image, we normalize V_i in order to obtain matrices R_i. Normalization here means that we divide each element of V_i by its number of updates (different elements are updated different number of times) and then linearly remap all obtained values into the interval $[0, 1]$.

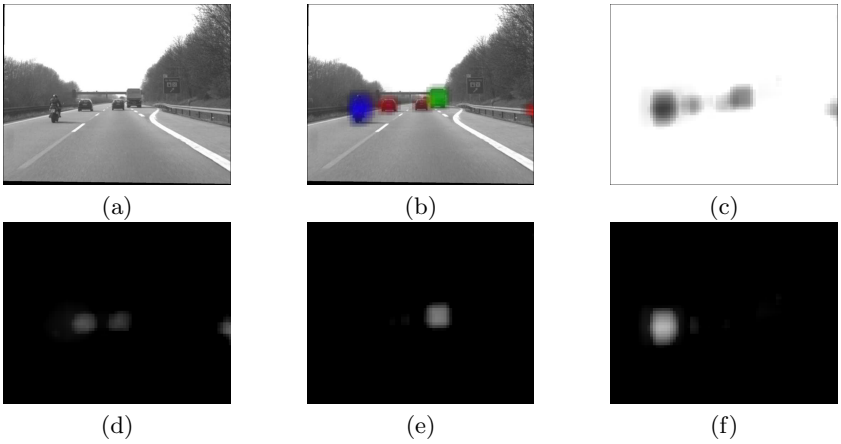

Fig. 8. Result of recognition for frame number 342 of motorway3 sequence. (a) Input grayscale image. (b) Input image overlaid with the output of the recognition stream (all the classes except background), showing, from left to right: motorcycle, two cars and truck. (c)–(f) Likelihood maps for the classes *background*, *cars*, *trucks* and *motorbikes* respectively. Pixel intensities correspond to the likelihood values.

To train LeNet, we have prepared a dataset of 64×64 grayscale images (approximately 67500 backgrounds, 24500 cars, 2500 motorbikes, 6200 trucks, 1900 bicycles, 78 buses, and 3500 pedestrians). Although this may seem an unbalanced dataset, it should reflect the proportion of different object classes present in real driving scenes, especially in rural ones. Furthermore, we have doubled the dataset by including horizontally flipped versions of all the samples. Images were taken mainly from publicly available object recognition

databases (LabelMe[3], VOC[4]). We have randomly permuted samples in the entire dataset and then splitted the latter up into training (90%) and testing (10%) sets. A stochastic version of the Levenberg-Marquardt algorithm with diagonal approximation of the Hessian [19] was used for LeNet training. Training was stopped after reaching a misclassification rate on the testing set of less than 1.5%. To increase the robustness of the classification, we have run the training procedure several times, every time by adding a small (2%) amount of uniform noise and by randomly changing the intensity (97–103%) of each training sample.

2.5 Visual streams fusion

Fusion of the visual streams for a particular frame is achieved in three steps.

1. Masking (elementwise multiplication) of the independent motion map I by the mask M of the most probable locations of the IMOs in the frame (see Fig. 9):
$$F_1(x, y) = I(x, y)M(x, y). \tag{2}$$
2. Masking of the previous result F_1 by the non-background map $(1 - R_0)$:
$$F_2(x, y) = F_1(x, y)(1 - R_0(x, y)). \tag{3}$$
3. Masking of the previous result F_2 by the likelihood maps $R_1, \ldots, R_6$ of each class, which results in six maps $L_1, \ldots, L_6$ (one for each class, except the background):
$$L_k(x, y) = F_2(x, y)R_k(x, y), \quad k = 1, \ldots, 6. \tag{4}$$

The first step is necessary for rejecting regions of the frame where the appearance of the IMOs is implausible. After the second step we obtain crucial information about regions which have been labeled as non-backgrounds (vehicles or pedestrians) and which, at the same time, contain independently moving objects. This information is represented as the saliency map F_2, which we will further use for IMO detection/description and in the tracking procedure. The third step provides us the information needed in the classification stage.

3 IMO Detection and Tracking

For detecting an IMO, we have used a simple technique based on the detection of the local maximas in the maps defined in (3). We have performed a spatio-temporal filtering (i.e. for i-th frame we apply smoothing of a three-dimensional array – a concatenation of the $(i-2)$-th, $(i-1)$-th, i-th, $(i+1)$-th

[3] http://labelme.csail.mit.edu/
[4] http://www.pascal-network.org/challenges/VOC/

Fig. 9. Matrix M, masking regions of possible IMO appearance in a frame. This matrix has been chosen heuristically based on the logical assumptions that in normal circumstances IMOs are not supposed to appear "in the sky" or beneath the ego-vehicle.

and $(i+2)$-th two-dimensional maps along the third time-dimension). Then we search for local maximas in the entire (i-th) filtered frame and consider them as the IMO centers $\mathbf{x}_k$ for this frame.

For tracking IMOs, we have introduced a parameter called *tracking score*. For a particular IMO, we increase this parameter when, in the next frame, only in a small neighborhood of the IMO center there is a good candidate for the considered IMO in the next frame, namely the IMO with the same class label, and approximately with the same properties (size, distance and relative speed in depth). Otherwise, the tracking score is decreased. An IMO survives while the tracking score is above a fixed threshold. The tracking score works as a momentum and allows the system to keep tracking an IMO even when there are no sufficient data in the next few frames.

4 Classification and description of the IMOs

As soon as we are able to detect IMOs, it becomes possible to classify them and to retrieve their properties (size, absolute speed in depth, relative speed in depth, time to contact, absolute acceleration, *etc*).

We define the *class* c_k of the k-th IMO as:

$$c_k = \arg \max_{1 \leq c \leq 6} \{L_c(\mathbf{x}_c)\}, \tag{5}$$

where $\mathbf{x}_k = (i_k, j_k)$ is the center of the k-th IMO (in image domain D) and L_c are the maps, defined in (4).

Let σ_k be the *size* of the k-th IMO. For σ_k estimation, we search for the spread of the appropriately scaled circular Gaussian blob, locally best fitting

L_{c_k} in $\mathbf{x}_k$. Analytically this can be expressed as a search for the argument of the first minimum of the function:

$$\Delta_k(\sigma) = \int_{D_k} \left| L_{c_k}(\mathbf{x}_k) e^{-||\mathbf{x}_k - \mathbf{x}||^2/\sigma^2} - L_{c_k}(\mathbf{x}) \right| d\mathbf{x}, \tag{6}$$

where D_k is a neighborhood of $\mathbf{x}_k$ in image D. In our simulations we have used $D_k = D$, but the choice of D_k could easily be optimized.

The IMO's *distance* estimation is a crucial point in the retrieval process. Using an averaged (in a small neighborhood of the IMO's center) disparity and known calibration parameters of the two cameras, we have computed the distance to the IMO. To compensate for instabilities in the distance estimations, we have used a robust linear regression based on the previous five estimates.

The present-day motor vehicles are being equipped with an increasing number of electronic devices, including control units, sensors, actuators, *etc.* All these devices communicate with each other over a data bus. During recording sessions, we have stored the egospeed provided by test car's speedometer.

The *relative speed in depth*, we estimated as the derivative (with respect to time) of the distance using robust linear regression based on the last five estimations of the distance. To estimate the *time to contact*, we have divided the averaged distance by the averaged relative speed in depth. Using the precise value of the ego-motion speed from the CAN-bus data, and simply by adding it to the relative speed in depth we have also obtained the *absolute speed in depth* of the considered IMO.

The derivative of the absolute speed in depth can be considered as an estimation of the *acceleration* (it is true only in the case when the ego-heading is collinear to the heading of the IMO). An example of IMO tracking and the retrieved properties is shown in Fig. 13.

5 LIDAR sensor data processing

The ACC system of the used test car was able to detect and track up to ten obstacles, when in the range of the LIDAR sensor. In addition to position, the ACC can also provide information about relative lateral extent and speed of the tracked obstacle.

5.1 LIDAR sensor setup

We used data recorded by the test car equipped with the LIDAR sensor manufactured by Hella KGaA Hueck & Co (see Table 4 for specifications). The sensor was mounted in the test car at 30 cm height above ground, with 18 cm from the front-end and 50 cm from the middle of the car to the driver's side (see Fig. 10). The ACC system analyzes raw LIDAR data and tracks up to 10 targets within a distance of up to 150 m. The tracking data are updated

and available for recording via the CAN-bus (Flex-ray) every 60 ms. Each tracked target is described by its distance, lateral position (left and right edges), relative velocity and acceleration.

Table 4. LIDAR sensor specifications

Sensor parameter	Value
Manufacturer	Hella KGaA Hueck & Co
Model	IDIS 1.0
Field of view	$12^\circ \times 4^\circ$ (horizontal×vertical)
Range	up to 200 m
Description	12 fixed horizontally distributed beams, each beam observes a $1^\circ \times 4^\circ$ angular cell

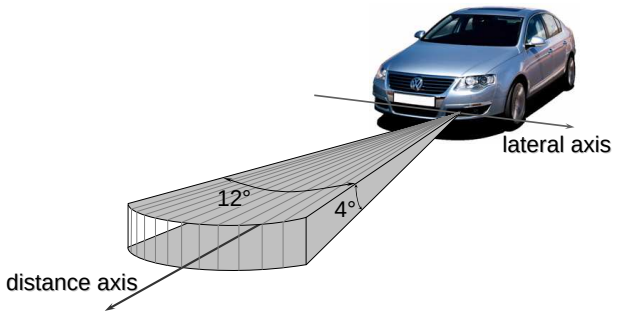

Fig. 10. ACC LIDAR configuration.

5.2 Ground plane estimation

The LIDAR provides the depth and lateral position of the detected obstacles. This information is not sufficient for the correct projection of the obstacles onto the video frame. In order to estimate the missing vertical components (in the frame domain) of the IMOs we assume that all IMOs are located near the dominant ground plane. Here we use a strong assumption of road planarity, which is not met in all driving scenarios and could introduce bias. However, in our model, the positions of the LIDAR-based obstacles are used only to verify (confirm) vision-based obstacles, so that the bias caused by the non-planarity of the road is to a large extend unimportant.

In order to estimate the ground plane, we estimate the *disparity plane*, then map the set of points from the disparity domain into a 3D world domain, and finally fit a plane through the projected set.

Before the disparity plane estimation, we intersect the disparity map with the predefined road mask (see Fig. 11, left panel). By this step, we filter out the majority of pixels which do not belong to the ground plane and are outliers in the disparity plane linear model:

$$\Delta : D = \alpha x + \beta y + \gamma, \tag{7}$$

where (x, y) are pixel coordinates and D is disparity.

Fig. 11. (Left) Predefined road mask. (Right) Example of the ground plane estimation. Besides the estimated ground plane, one can see the estimated horizon line and the points used for the ground plane estimation (see text).

The disparity plane parameters α, β and γ are estimated using IRLS (Iteratively Reweighted Least-Squares) with weight function proposed by Beaton and Tukey [1] and tuning parameter $c = 4.6851$. 7 iterations.

For the ground plane parameters estimation, we choose a set of nine points (3×3 lattice) in the lower half of the frame (see Fig. 11, right panel). Disparities for these points are determined using the estimated disparity plane (7). Given the disparities and camera calibration data, we project the selected points into a 3D world coordinate system. In addition, we add two so-called *stabilization points* which correspond to the points where the front wheels of the test car are supposed to touch the road surface. For the inverse projection of the stabilization points, we use parameters of the *canonical disparity plane*: it is a disparity plane which corresponds to the horizontal ground plane observed by cameras in a quiescent state. The parameters of the canonical disparity plane and positions of the stabilization points were obtained based on the test car geometry and camera setup position and orientation in the test car. The full set of 11 points is then used for IRLS fitting of the ground plane in a world coordinate system:

$$\pi : aX + bY + cZ + d = 0, \tag{8}$$

where (X, Y, Z) are pixel coordinates in the 3D world coordinate system connected to the left camera. Here were assume that $a^2 + b^2 + c^2 = 1$ (otherwise

one can divide all coefficients by $\sqrt{a^2+b^2+c^2}$) and $b > 0$. In this case vector $\mathbf{n} = (a, b, c)^T$ represents the normal unity vector of the ground plane and coefficient d represents the distance from the camera to the ground plane. During the disparity plane estimation, we use the estimation from the previous frame for weight initialization in IRLS; for the first frame, for the same purpose, we use the parameters of the canonical disparity plane. We assume that the ground plane is estimated correctly if the following conditions are met:

$$\|\mathbf{n}_t - \mathbf{n}_0\| < \theta_0 \text{ and } \|\mathbf{n}_t - \mathbf{n}_{t-1}\| < \theta_1, \tag{9}$$

where $\mathbf{n}_k$ is normal vector for $k-th$ frame, and $\mathbf{n}_0$ is canonical normal vector. Thresholds $\theta_0 = 0.075$ and $\theta_1 = 0.015$ were chosen empirically. If the estimated ground plane does not satisfy (9), the previous estimation is used.

5.3 LIDAR obstacles projection

Projection of the LIDAR-based obstacles into the (left) frame is based on the ground plane position, the obstacle positions, the camera projective matrix (from calibration data) and the position and orientation of the LIDAR sensor with respect to the camera. Only the height of the obstacles is not available. We have set the height of all the obstacles to a fixed value of 1.5 m. The result of the LIDAR obstacles projection is shown in Fig. 12.

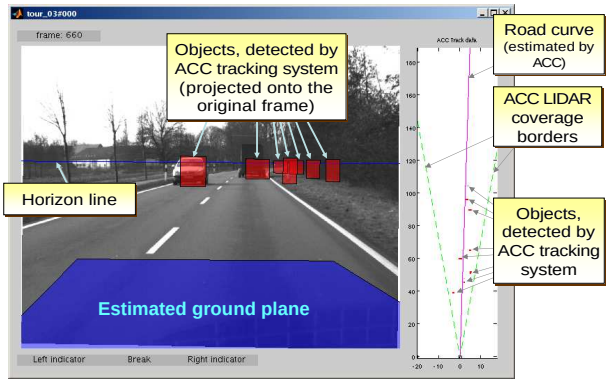

Fig. 12. ACC obstacles projection. Left part contains the grayscale version of current frame, overlaid by the horizon line, the ground plane segment and projected ACC (LIDAR) obstacles. Right part represents obstacles 2D range profile, provided by ACC system.

6 Vision and LIDAR fusion

The fusion of the vision-based IMOs with LIDAR-based obstacles is based on a simple matching process.

1. For the current IMO I_k, we look for candidates from the LIDAR obstacles O_l by means of the high intersection ratio:
$$r_{kl} = \#(I_k \cap O_l)/\#(I_k), \tag{10}$$
where $\#(\cdot)$ is number of pixels of the set in the brackets. If ratio $r_{kl} > 0.5$, then obstacle O_l is an IMO I_k candidate and considered for further verification. If all obstacles were rejected, IMO I_k is not updated and process continues from step 4.
2. All the obstacles O_{k_m} with distances d_{k_m} satisfying the following condition:
$$\frac{|d_{k_m} - d_k^*|}{d_k^*} > 0.15, \tag{11}$$
where d_k^* denotes the distance of the IMO I_k, are rejected. Like in the previous step, if all obstacles were rejected, IMO I_k is not updated and the process continues from step 4.
3. Among the remaining obstacles, we choose the best matching candidate O_{k_i} for the IMO I_k with minimal depth deviation $|d_{k_i} - d_k^*|$. Distance, relative velocity and acceleration of IMO I_k are updated using corresponding values of the obstacle O_{k_i}. The absolute velocity of the IMO I_k is re-estimated in accordance with the new value of the relative speed. The obstacle O_{k_i} is eliminated from the search process. If all the obstacles were rejected, IMO I_k is not updated.
4. The process finishes if all IMOs are checked, otherwise the next IMO is selected for matching and the process continues from step 1.

7 Results

Due to the lack of ground truth benchmarks, systems similar to the proposed one are tested mainly in a qualitative sense. For the evaluation of the presented system, we have used two complex real-world video sequences (different from the training sequences). ACC LIDAR tracking data has been provided only for one of them. Nevertheless, even without this important information the system has shown the ability to detect, track and describe IMOs (see Fig. 13) relying only on the data from the vision sensor and the IMU (speedometer). This testing scenario is important because during rapid turns or when the IMOs are out of reach of the LIDAR sensor, the system has to rely on the vision sensor only. In this case, the quality of the properties estimation suffers from the noise presented in the visual cues.

The experiments with the ACC LIDAR tracking data (see Fig. 14) have shown a significant improvement of the accuracy of the IMOs properties extraction especially in low curvature driving scenarios (when the ACC latencies have a negligible influence on the LIDAR-based obstacle localization).

One of the most important results is the fact that both networks have shown to be acceptable of generating qualitative results even on completely

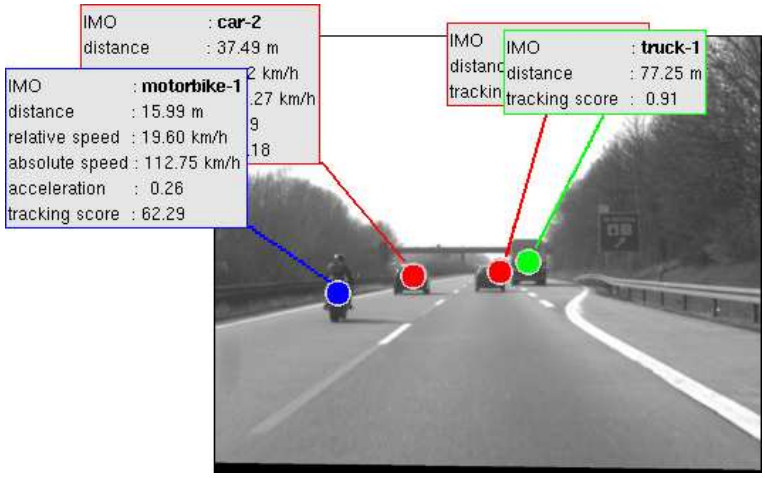

Fig. 13. Vision-based IMO detection, classification, description and tracking result.

new testing sequences, which attests to the generalization ability of the used networks.

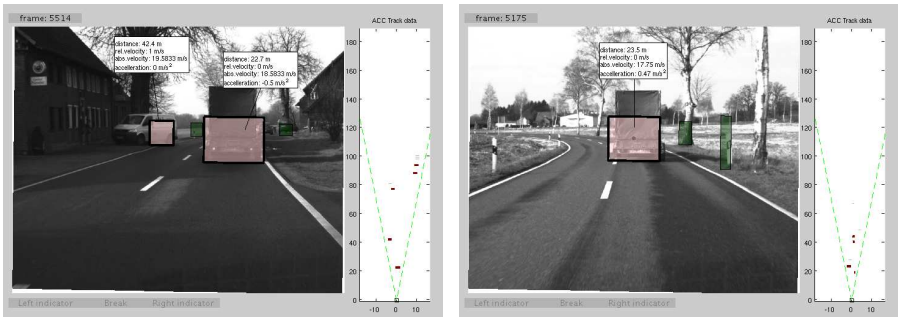

Fig. 14. Results of the final fusion. Light (pink) bars with thick edges represent the detected IMOs, whereas the LIDAR obstacles rejected by the fusion procedure are shown as dark (green) bars.

8 Conclusions and future steps

A high level sensor fusion model for IMO detection, classification and tracking has been proposed. The model incorporates three independent sensors: vision, LIDAR and speedometer. Vision plays the most important role in the model, whereas LIDAR data are used for confirming the IMO detection and

for updating the IMO properties. The speedometer is used only for the IMOs absolute speed in depth estimation.

The existing model is still not a real-time system, but we see a number of ways to increase its speed. Both visual streams of the model have feed-forward architectures, which can be easily implemented in hardware such as Field-Programmable Gate Arrays (FPGAs). Moreover, as far as the streams are independent, they can be implemented as separate FPGAs, working in parallel. In order to speed up the entire model, we propose to switch from LeNet-based object recognition to faster and more task-specific recognition paradigm (e.g. [32] or [20]). Another way to increase the speed of the model could be the transition from an MLP-based fusion of the visual cues to a hard-coded fusion of the visual cues with egomotion (e.g. [22]). As another future step of the model development, we envisage the incorporation of KF-based approaches [13, 21] for IMO tracking.

Finally, as more abstract descriptions of the IMOs are generated, this is expected to facilitate the development of models of driving behavior, in response to IMOs, which is one of the goals of the DRIVSCO project[5]. Indeed, *stopping*, *following*, *lane changing*, and so on, are important aspects of driving behavior, and which are key elements of driver-assistance systems that support collision free driving and increased traffic safety in general. These are heavily researched topics in academia, research institutes and the automotive industry.

9 Acknowledgments

The first author is supported by the European Commission (NEST-2003-012963). The second author is supported by the Excellence Financing program (EF 2005) and the CREA Financing program (CREA/07/027) of the K.U.Leuven, the Belgian Fund for Scientific Research – Flanders (G.0248.03, G.0234.04), the Flemish Regional Ministry of Education (Belgium) (GOA 2000/11), the Belgian Science Policy (IUAP P5/04), and the European Commission (NEST-2003-012963, STREP-2002-016276, and IST-2004-027017).

References

[1] Beaton, A., Tukey, J.: The fitting of power series, meaning polynomials, illustrated on band-spectroscopic data. Technometrics **16**(2), 147–185 (1974)

[2] Becker, J., Simon, A.: Sensor and navigation data fusion for an autonomous vehicle. Intelligent Vehicles Symposium, 2000. IV 2000. Proceedings of the IEEE pp. 156–161 (2000)

[5] http://www.pspc.dibe.unige.it/drivsco/

[3] Bertozzi, M., Broggi, A., Cellario, M., Fascioli, A., Lombardi, P., Porta, M.: Artificial vision in road vehicles. Proceedings of the IEEE **90**(7), 1258–1271 (2002)
[4] Bertozzi, M., Broggi, A., Fascioli, A.: Vision-based intelligent vehicles: State of the art and perspectives. Robotics and Autonomous Systems **32**(1), 1–16 (2000)
[5] Blanc, C., Trassoudaine, L., Le Guilloux, Y., Moreira, R.: Track to track fusion method applied to road obstacle detection. Procedings of the Seventh International Conference on Information Fusion (2004)
[6] Chumerin, N., Van Hulle, M.: An approach to on-road vehicle detection, description and tracking. Proceedings of the 2007 17th IEEE Signal Processing Society Workshop on Machine Learning for Signal Processing (2007). (in press)
[7] Dasarathy, B.: Sensor fusion potential exploitation-innovative architectures and illustrative applications. Proceedings of the IEEE **85**(1), 24–38 (1997)
[8] Fang, Y., Masaki, I., Horn, B.: Depth-based target segmentation for intelligent vehicles: fusion of radar and binocular stereo. Intelligent Transportation Systems, IEEE Transactions on **3**(3), 196–202 (2002)
[9] Gandhi, T., Trivedi, M.: Vehicle surround capture: Survey of techniques and a novel omni video based approach for dynamic panoramic surround maps. Intelligent Transportation Systems, IEEE Transactions on **7**(3), 293–308 (2006)
[10] Hall, D., Llinas, J.: An introduction to multisensor data fusion. Proceedings of the IEEE **85**(1), 6–23 (1997)
[11] Handmann, U., Lorenz, G., Schnitger, T., Seelen, W.: Fusion of different sensors and algorithms for segmentation. IV'98, IEEE International Conference on Intelligent Vehicles 1998 pp. 499–504 (1998)
[12] Hofmann, U., Rieder, A., Dickmanns, E.: Radar and vision data fusion for hybrid adaptive cruise control on highways. Machine Vision and Applications **14**(1), 42–49 (2003)
[13] Julier, S., Uhlmann, J.: A new extension of the kalman filter to nonlinear systems. Int. Symp. Aerospace/Defense Sensing, Simul. and Controls **3** (1997)
[14] Kalman, R.: A new approach to linear filtering and prediction problems. Journal of Basic Engineering **82**(1), 35–45 (1960)
[15] Kastrinaki, V., Zervakis, M., Kalaitzakis, K.: A survey of video processing techniques for traffic applications. Image and Vision Computing **21**(4), 359–381 (2003)
[16] Kato, T., Ninomiya, Y., Masaki, I.: An obstacle detection method by fusion of radar and motion stereo. Intelligent Transportation Systems, IEEE Transactions on **3**(3), 182–188 (2002)
[17] Labayrade, R., Royere, C., Gruyer, D., Aubert, D.: Cooperative fusion for multi-obstacles detection with use of stereovision and laser scanner. Autonomous Robots **19**(2), 117–140 (2005)

[18] Laneurit, J., Blanc, C., Chapuis, R., Trassoudaine, L.: Multisensorial data fusion for global vehicle and obstacles absolute positioning. Intelligent Vehicles Symposium, 2003. Proceedings. IEEE pp. 138–143 (2003)
[19] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. pp. 2278–2324 (1998)
[20] Leibe, B., Schiele, B.: Scale invariant object categorization using a scale-adaptive mean-shift search. DAGM'04 pp. 145–153 (2004)
[21] van der Merwe, R.: Sigma-point kalman filters for probabilistic inference in dynamic state-space models. Ph.D. thesis, University of Stellenbosch (2004)
[22] Pauwels, K., Van Hulle, M.: Segmenting independently moving objects from egomotion flow fields. Isle of Skye, Scotland (2004)
[23] Pauwels, K., Van Hulle, M.: Optic flow from unstable sequences containing unconstrained scenes through local velocity constancy maximization. pp. 397–406. Edinburgh (2006)
[24] Pohl, C.: Review article multisensor image fusion in remote sensing: concepts, methods and applications. International Journal of Remote Sensing **19**(5), 823–854 (1998)
[25] Sabatini, S., Gastaldi, G., Solari, F., Diaz, J., Ros, E., Pauwels, K., Van Hulle, M., Pugeault, N., Krueger, N.: Compact and accurate early vision processing in the harmonic space. Barcelona (2007)
[26] Sergi, M.: Bus rapid transit technologies: A virtual mirror for eliminating vehicle blind zones. University of Minnesota ITS Institute Final Report (2003)
[27] Sole, A., Mano, O., Stein, G., Kumon, H., Tamatsu, Y., Shashua, A.: Solid or not solid: vision for radar target validation. Intelligent Vehicles Symposium, 2004 IEEE pp. 819–824 (2004)
[28] Steux, B., Laurgeau, C., Salesse, L., Wautier, D.: Fade: a vehicle detection and tracking system featuring monocular color vision and radar data fusion. Intelligent Vehicle Symposium, 2002. IEEE **2** (2002)
[29] Stiller, C., Hipp, J., Rossig, C., Ewald, A.: Multisensor obstacle detection and tracking. Image and Vision Computing **18**(5), 389–396 (2000)
[30] Sun, Z., Bebis, G., Miller, R.: On-road vehicle detection: a review. Pattern Analysis and Machine Intelligence, IEEE Transactions on **28**(5), 694–711 (2006)
[31] Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G., et al.: Stanley: The robot that won the darpa grand challenge. Journal of Field Robotics **23**(9), 661–692 (2006)
[32] Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. Proc. CVPR **1**, 511–518 (2001)
[33] Wald, L.: Some terms of reference in data fusion. IEEE Transactions on Geoscience and Remote Sensing **37**(3) (1999)

Index